IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	Boucher, et al.	Serial No.:	Unknown
Filed:	October 3, 2003	Examiner:	Unknown
Docket No.	ALA-008H	GAU:	Unknown
Assignee:	Alacritech Inc.		
Title:	METHOD AND APPARATUS FOR DYNAMIC PACKET BATCHING WITH A HIGH PERFORMANCE NETWORK INTERFACE		

Mail Stop PATENT APPLICATION
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

October 3, 2003

REQUEST TO PROVOKE INTERFERENCE

Sir:

Applicants hereby seek to have an interference declared between the present application and U.S. Patent No. 6,483,804, issued November 12, 2002.

U.S. Patent No. 6,483,804 to Muller et al. issued from U.S. application serial number 09/260,324, filed March 1, 1999.

The present application claims the benefit under 35 U.S.C. §120 of (is a continuation of) U.S. Patent Application Serial No. 10/005,536, filed November 7, 2001, which in turn claims the benefit under 35 U.S.C. §120 of (is a continuation of) U.S. patent application serial number 09/384,792, filed August 27, 1999, now U.S. Patent No. 6,434,620, which in turn: 1) claims the benefit under 35 U.S.C. §119 of provisional patent application serial number 60/098,296, filed August 27, 1998, 2) claims the benefit under 35 U.S.C. §120 of (is a continuation-in-part of) U.S. patent application serial number 09/067,544, filed April 27, 1998,

now U.S. Patent No. 6,226,680, and 3) claims the benefit under 35 U.S.C. §120 of (is a continuation-in-part of) U.S. patent application serial number 09/141,713, filed August 28, 1998, now U.S. Patent No. 6,389,479.

U.S. Patent No. 6,226,680 and U.S. Patent No. 6,389,479 both claim the benefit under 35 U.S.C. §119 of provisional patent application serial number 60/061,809, filed October 14, 1997. The present application also claims the benefit under 35 U.S.C. §120 of (is a continuation-in-part of) U.S. Patent Application Serial No. 09/464,283, filed December 15, 1999, now U.S. Patent No. 6,427,173, and claims the benefit under 35 U.S.C. §120 of (is a continuation-in-part of) U.S. Patent Application Serial No. 09/514,425, filed February 28, 2000, now U.S. Patent No. 6,427,171.

Claims 1-27 of the present application are copied from Claims 1-27 of U.S. Patent No. 6,483,804, respectively.

REQUIREMENTS OF 37 CFR 607:

Per rule 607(a)(1), the patent is U.S. Patent No. 6,483,804.

Per rule 607(a)(2), Applicants submit Counts 1-27 as set forth below.

Count #1 (corresponds exactly to Claim 1 of the '804 patent):

A method of identifying multiple packets in a communication flow between a source entity and a destination entity, comprising:

storing a first flow identifier of a first packet received from a source entity for a destination entity, wherein said first flow identifier comprises an identifier of the source entity and an identifier of the destination entity;

storing said first packet in a packet memory for transfer toward the destination entity;

storing a second flow identifier of a second packet;

storing said second packet in said packet memory;

determining whether said first flow identifier matches said second flow identifier;

storing a first indicator in the destination entity if a first communication flow identified by said first flow identifier comprises said second packet; and

storing a second indicator in the destination entity if said first packet is the only packet stored in the packet memory that is part of said first communication flow.

Count #2 (corresponds to Claim 2 of the '804 patent):

A method of identifying multiple packets in a communication flow between a source entity and a destination entity, comprising:

storing a first flow identifier of a first packet received from a source entity for a destination entity, wherein said first flow identifier comprises an identifier of the source entity and an identifier of the destination entity;

storing said first packet in a packet memory for transfer toward the destination entity;

storing a second flow identifier of a second packet;

storing said second packet in said packet memory;

determining whether said first flow identifier matches said second flow identifier;

storing a first indicator in the destination entity if a first communication flow identified by said first flow identifier comprises said second packet;

storing a second indicator in the destination entity if said first packet is the only packet stored in the packet memory that is part of said first communication flow; and

prior to said storing a first flow identifier, parsing said first packet to retrieve said identifier of the source entity and said identifier of the destination entity.

Count #3 (corresponds exactly to Claim 3 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet.

Count #4 (corresponds to Claim 4 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device;

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet; and

transferring said second packet to said host computer;

wherein said host computer is configured to collectively process a header portion of said first packet and a header portion of said second packet in accordance with said communication protocol.

Count #5 (corresponds to Claim 5 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet;

wherein said identifying comprises:

receiving a flow key generated by concatenating an identifier of the source entity and an identifier of the destination entity;

wherein said first flow identifier comprises said flow key.

Count #6 (corresponds to Claim 6 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet;

wherein said identifying comprises:

receiving an index of said first communication flow in a flow database;

wherein said first flow identifier comprises said index.

Count #7 (corresponds to Claim 7 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet;

wherein said determining comprises comparing said first flow identifier with a second flow identifier associated with a second packet received at said communication device.

Count #8 (corresponds to Claim 8 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet;

wherein said determining comprises comparing said first flow identifier with a second flow identifier associated with a second packet received at said communication device;

storing said first flow identifier in a flow memory; and

storing said second flow identifier in said flow memory; and

comparing said stored first flow identifier and said stored second flow identifier.

Count #9 (corresponds to Claim 9 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet;

wherein said determining comprises comparing said first flow identifier with a second flow identifier associated with a second packet received at said communication device;

storing said first flow identifier in a flow memory; and

storing said second flow identifier in said flow memory; and

comparing said stored first flow identifier and said stored second flow identifier;

wherein said flow memory is an associative memory in said communication device.

Count #10 (corresponds to Claim 10 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device;

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet; and

storing said first packet in a packet memory.

Count #11 (corresponds to Claim 11 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device;

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet; and

storing said first packet in a packet memory;

wherein said determining comprises comparing said first flow identifier configured to identify said first communication flow with a second flow identifier configured to identify a second communication flow comprising a packet stored in said packet memory.

Count #12 (corresponds to Claim 12 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device;

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet; and

informing said host computer of said transfer of said first packet;

wherein said informing comprises configuring an indicator in a host memory, and said indicator is configured to indicate that said host computer should delay processing said first packet until said second packet is transferred to said host computer.

Count #15 (corresponds to Claim 15 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device;

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet; and

informing said host computer of said transfer of said first packet;

wherein said informing comprises configuring an indicator in a host memory, and said indicator indicates that said host computer should not delay processing said first packet.

Count #16 (corresponds exactly to Claim 16 of the '804 patent):

A method of transferring a packet from a network interface to a host computer, comprising:

receiving a first packet at a network interface;

storing said first packet in a packet memory;

receiving a first flow identifier configured to identify a communication flow comprising said first packet;

storing said first flow identifier in a flow memory;

searching said flow memory for a second packet in said communication flow received at the network interface after said first packet;

transferring said first packet to said host computer; and

configuring an indicator in a host memory to indicate whether processing of said first packet by said host computer should be delayed to await transfer of said second packet to said host memory.

Count #17 (corresponds to Claim 17 of the '804 patent):

A method of transferring a packet from a network interface to a host computer, comprising:

receiving a first packet at a network interface;

storing said first packet in a packet memory;

receiving a first flow identifier configured to identify a communication flow comprising said first packet;

storing said first flow identifier in a flow memory;

searching said flow memory for a second packet in said communication flow received at the network interface after said first packet;

transferring said first packet to said host computer; and

configuring an indicator in a host memory to indicate whether processing of said first packet by said host computer should be delayed to await transfer of said second packet to said host memory;

wherein said generating comprises:

receiving an index of said communication flow in a flow database;

wherein said flow identifier comprises said index.

Count #18 (corresponds to Claim 18 of the '804 patent):

A method of transferring a packet from a network interface to a host computer, comprising:

receiving a first packet at a network interface;

storing said first packet in a packet memory;

receiving a first flow identifier configured to identify a communication flow comprising said first packet;

storing said first flow identifier in a flow memory;

searching said flow memory for a second packet in said communication flow received at the network interface after said first packet;

transferring said first packet to said host computer; and

configuring an indicator in a host memory to indicate whether processing of said first packet by said host computer should be delayed to await transfer of said second packet to said host memory;

wherein said receiving comprises:

receiving a flow key comprising an identifier of a source of said first packet and an identifier of a destination of said first packet;

wherein said flow identifier comprises said flow key.

Count #19 (corresponds to Claim 19 of the '804 patent):

A method of transferring a packet from a network interface to a host computer, comprising:

receiving a first packet at a network interface;

storing said first packet in a packet memory;

receiving a first flow identifier configured to identify a communication flow comprising said first packet;

storing said first flow identifier in a flow memory;

searching said flow memory for a second packet in said communication flow received at the network interface after said first packet;

transferring said first packet to said host computer; and

configuring an indicator in a host memory to indicate whether processing of said first packet by said host computer should be delayed to await transfer of said second packet to said host memory;

wherein said packet memory comprises said flow memory.

Count #20 (corresponds to Claim 20 of the '804 patent):

A method of transferring a packet from a network interface to a host computer, comprising:

receiving a first packet at a network interface;

storing said first packet in a packet memory;

receiving a first flow identifier configured to identify a communication flow comprising said first packet;

storing said first flow identifier in a flow memory;

searching said flow memory for a second packet in said communication flow received at the network interface after said first packet;

transferring said first packet to said host computer; and

configuring an indicator in a host memory to indicate whether processing of said first packet by said host computer should be delayed to await transfer of said second packet to said host memory;

wherein said configuring comprises:

storing a first indicator in a host memory if said communication flow comprises said second packet; and

storing a second indicator in said host memory if said first packet is the only packet in said packet memory that is part of said communication flow.

Count #21 (corresponds exactly to Claim 21 of the '804 patent):

A computer system for processing a packet received from a network interface, comprising:

a network interface configured to receive a first packet from a network and transfer said first packet to a host computer memory, said network interface comprising:

a packet memory configured to store said first packet;

a flow memory for storing a first flow number associated with said first packet, wherein said first flow number is configured to identify a communication flow comprising said first packet;

a packet batcher configured to determine whether the communication flow includes a second packet stored in said packet memory after said first packet; and

a notifier configured to:

store a first code in a host indicator if said packet memory includes the second packet; and

store a second code in said host indicator if said packet memory does not include the second packet; and

a processor for processing a header portion of said first packet.

Count #22 (corresponds exactly to Claim 22 of the '804 patent):

A computer readable storage medium storing instructions that, when executed by a computer, cause the computer to perform a method of transferring a packet from a network interface to a host computer, the method comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet.

Count #23 (corresponds exactly to Claim 23 of the '804 patent):

A processor readable storage medium containing a data structure configured to store information concerning a packet to be transferred from a network interface to a host computer, the data structure including one or more entries, each entry comprising:

a flow number configured to identify a communication flow comprising a first packet received at the network interface from a source entity for a destination entity associated with the host computer; and

a validity indicator configured to provide:

a first indication if said first packet is ready for transfer to the host computer; and

a second indication if said first packet is a control packet;

wherein said data structure is searched for a second entry containing said flow number when said first packet is transferred to the host computer to determine if said communication flow also comprises a second packet received at the network interface after said first packet.

Count #24 (corresponds to Claim 24 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet;

wherein said identifying comprises:

parsing said first packet to retrieve an identifier of the source entity and an identifier of the destination entity; and

combining said source entity identifier and said destination entity identifier to form said first flow identifier.

Count #25 (corresponds exactly to Claim 25 of the '804 patent):

A communication interface, comprising:

a header parser configured to parse a header of a first packet received at the communication interface, wherein the first packet was issued from a source entity for a destination entity;

a flow database configured to facilitate management of a communication flow comprising the first packet, the flow database comprising:

a flow key configured to identify the communication flow using identifiers of the source entity and the destination entity;

an activity indicator configured to indicate a recency with which a packet in the communication flow has been received; and

a validity indicator for indicating whether the communication flow is valid;

a code generator configured to generate an operation code for the first packet, to facilitate forwarding of the first packet toward the destination entity; and

a packet batching module configured to determine whether a second packet received at the communication interface is part of the communication flow.

Count #26 (corresponds exactly to Claim 26 of the '804 patent):

A method of processing a packet through a communication interface, the method comprising:

receiving a first packet from a network, wherein the first packet is part of a communication flow between a source entity and a destination entity;

determining whether a header portion of the first packet conforms to one of a set of communication protocols;

assembling a flow identifier to identify the communication flow, wherein said flow identifier comprises a source entity identifier and a destination entity identifier;

updating a flow database configured to facilitate management of communication flows through the communication interface, wherein said updating comprises:

configuring a flow activity indicator associated with the communication flow to reflect receipt of the first packet; and

configuring a flow validity indicator associated with the communication flow to indicate that the communication flow is valid;

assigning an operation code to the first packet, said operation code indicating whether a portion of data in the first packet is reassembleable with another portion of data in another packet in the communication flow; and

determining whether a second packet received at the communication interface is part of the communication flow.

Count #27 (corresponds to Claim 27 of the '804 patent):

A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:

receiving a first packet at a communication device;

identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device;

transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet;

storing a first indicator in the host computer if said first communication flow comprises said second packet; and

storing a second indicator in the host computer if said first packet is the only packet stored in the communication device that is part of said communication flow.

Per rule 607(a)(3), the claims in the patent (6,483,804) correspond to the counts as set forth in the table below. Per rule 607(a)(4), the claims in the present application correspond to the counts as set forth in the table below:

Count Number	Corresponding Claim in USP 6,483,804	Corresponding Claim in the present application
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	10	10
11	11	11
12	12	12
13	13	13
14	14	14
15	15	15
16	16	16
17	17	17
18	18	18
19	19	19
20	20	20
21	21	21
22	22	22
23	23	23
24	24	24
25	25	25
26	26	26
27	27	27

Per rule 607(a)(5), application of the terms of Claims 1-27 to the disclosure of the above-identified application is set forth in the claims chart below.

USP 6,453,360 Claim Element	Support in Alacritech Application Serial No. 60/098,296 (Page #: Line #)
1. A method of identifying multiple packets in a communication flow between a source entity and a destination entity, comprising:	(5:22-24) "So the next question is what to do about the ***network packets*** that do not ***fit our criteria***. The answer is to use two modes of operation: One in which the ***network frames*** are processed on the INIC through TCP and one in which the card operates like a typical dumb NIC."
storing a first flow identifier of a first packet received from a source entity for a destination entity, wherein said first flow identifier comprises an identifier of the source entity and an identifier of the destination entity;	(6:30-31) "When a ***frame is received*** by the ***INIC***, it must verify it completely before it even determines whether it belongs to one of its CCBs or not." (6:36-38) "The header is fully parsed by hardware and its type is summarized in a single ***status word***. The checksum is also verified automatically in hardware, and a ***hash*** key is created out of the ***IP addresses and TCP ports*** to expedite CCB lookup." (71:14-15) "These ***status words*** are generated by the INIC hardware and ***placed in front of the receive frame***."
storing said first packet in a packet memory for transfer toward the destination entity;	(85:26-27) "As ***frames are received***, they are ***placed into 2K-byte DRAM buffers*** by the Receive hardware sequencer, along with 16 bytes of the above frame status."
storing a second flow identifier of a second packet;	See above.
storing said second packet in said packet memory;	See above.
determining whether said first flow identifier matches said second flow identifier;	(86:1-6) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***. This ***hash*** is used to index directly into a ***hash table*** on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The microcode uses the ***hash*** to ***determine if a CCB exists on the INIC for this frame***."
storing a first indicator in the destination entity if a first communication flow identified by said first flow identifier comprises said second packet; and	(86:7-8) "If a ***match is found***, then the frame will be ***processed against the CCB by the INIC***." (5:27-28) "In the fast path case, ***network data is given to the host*** after the headers have been processed and stripped."

storing a second indicator in the destination entity if said first packet is the only packet stored in the packet memory that is part of said first communication flow.	(5:25-27) "In the slow-path case, ***network frames are handed to the system at the MAC layer*** and passed up through the host protocol stack like any other network frame." (83:8-9) "When the INIC receives a frame, one of its immediate tasks is to determine if the frame is for a CCB that it controls. If not, ***the frame is passed to the host*** on what is termed the slow path."
2. The method of claim 1, further comprising, prior to said storing a first flow identifier, parsing said first packet to retrieve said identifier of the source entity and said identifier of the destination entity.	(6:36-38) "The header is fully ***parsed*** by hardware and its type is summarized in a single status word. The checksum is also verified automatically in hardware, and a hash key is created out of the ***IP addresses and TCP ports*** to expedite CCB lookup." (85:15-28) "Much ***header validation*** is implemented in hardware in conjunction with MAC processing by scanning the data as it flies by. The following tests are performed and appropriate status is generated by the ***Receive hardware sequencer***. • MAC header: determine if Ethernet/802.3, if MC/BC, if it matches our MAC address A or B, determine the network protocol, flag if not a MAC status of "good packet." • Network header: determine if header checksum is valid, header length is valid (e.g. IP >= 5), network length > header length, what the transport protocol is, if there is any fragmentation or network options, ***destination network address*** (one of ours?) • Transport header: checksum is valid (incl pseudo-header if relevant), header length is valid (e.g. TCP >= 5), length is valid, ***what is the session layer protocol*** (e.g. SMB, HTTP or FTP data), are there any transport flags set (e.g. FIN/SYN/URG/RST bits), and any options present."
3. A method of identifying one or more packets in a communication flow between a source entity and a destination entity, comprising:	(5:22-24) "So the next question is what to do about the ***network packets*** that do not ***fit our criteria***. The answer is to use two modes of operation: One in which the ***network frames*** are processed on the INIC through TCP and one in which the card operates like a typical dumb NIC."
receiving a first packet at a communication device;	(6:30-31) "When a ***frame is received*** by the ***INIC***, it must verify it completely before it even determines whether it belongs to one of its CCBs or not."
identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;	(6:10-12) "A ***CCB*** is a structure that contains the entire ***context*** associated with a ***connection***. This includes the ***source and destination IP addresses and source and destination TCP ports*** that ***define the connection***. (4:16-19) "The ... ***context*** is ... ***identified by the IP source and destination addresses and TCP source and destination ports***." (6:36-38) "The header is fully parsed by hardware and its type is summarized in a single ***status word***. The checksum is also verified automatically in hardware, and a ***hash key*** is created out of the ***IP addresses and TCP ports*** to expedite ***CCB*** lookup."

	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***. This ***hash*** is used to index directly into a hash table on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The microcode uses the hash to ***determine if a CCB exists on the INIC for this frame***. It does this by following this chain from the hash table entry, and for each chained header table entry, ***comparing its source and destination addresses and ports with those of the frame***."
determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***. This ***hash*** is used to index directly into a hash table on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The microcode uses the hash to ***determine if a CCB exists on the INIC for this frame***. It does this by following this chain from the hash table entry, and for each chained header table entry, ***comparing its source and destination addresses and ports with those of the frame***."
transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet.	(5:25-27) "In the slow-path case, ***network frames are handed to the system at the MAC layer and passed up through the host protocol stack*** like any other network frame." (5:7-8) "As noted above, we do not support fragmented TCP segments on the initial INIC configuration."
4. The method of claim 3, further comprising:	See above.
transferring said second packet to said host computer;	See above.
wherein said host computer is configured to collectively process a header portion of said first packet and a header portion of said second packet in accordance with said communication protocol.	(5:17-18) "W. Richard Stevens and Gary Write in Volume 2 of their book "TCP/IP Illustrated", which is incorporated by reference herein." Stevens et al. notes, under the heading 10.6 ip_reass Function, beginning on page 286: "The ***IP header*** of incoming ***IP packets*** is converted to an ipasfrag structure (Figure 10.14) before it is ***placed on a reassembly list***. Ip_reass ***collects fragments*** for a ***particular datagram*** on a circular doubly linked list joined by the ipf_next and ipf_prev members." (See also Stevens, page 286, line 12-13) "If a ***complete datagram*** is available after ***reassembly processing***, it is passed up to the appropriate transport protocol by ipintr (Figure 8.15)."
5. The method of claim 3, wherein said identifying comprises:	See above.

receiving a flow key generated by concatenating an identifier of the source entity and an identifier of the destination entity;	(164:14-16) "CtxHsh The 8-bit ***context-hash generated by exclusive-oring all bytes of the IP source address, IP destination address, transport source port and the transport destination port.***" (86:1-2) "The ***receive sequencer has already generated*** a hash based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports.***"
wherein said first flow identifier comprises said flow key.	(6:36-38) "The header is fully parsed by hardware and its type is summarized in a single status word. The checksum is also verified automatically in hardware, and a ***hash key*** is created out of the ***IP addresses and TCP ports*** to expedite CCB lookup." (86:1-3) "The ***receive sequencer has already generated*** a hash based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports.***
6. The method of claim 3, wherein said identifying comprises:	
receiving an index of said first communication flow in a flow database; wherein said first flow identifier comprises said index.	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***. This ***hash*** is used to ***index*** directly into a ***hash table*** on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The microcode uses the hash to determine if a CCB exists on the INIC for this frame. It does this by following this chain from the hash table entry, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame."
7. The method of claim 3, wherein said determining comprises comparing said first flow identifier with a second flow identifier associated with a second packet received at said communication device.	(86:5-9) "The microcode uses the ***hash*** to determine if a CCB exists on the INIC for this frame. It does this by following this chain from the ***hash table*** entry, and for ***each chained header table entry, comparing*** its source and destination addresses and ports with those of the frame. If a match is found, then the frame will be processed against the CCB by the INIC. If not, then the frame is sent for slow-path processing."
8. The method of claim 7, wherein said determining further comprises:	See above.
storing said first flow identifier in a flow memory; and	(86:1-5) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This ***hash*** is used to index directly into a ***hash table*** on the INIC that points to entries in a ***CCB header table***. The header table entries are chained on the ***hash table*** entry. (6:10-16) "This introduces the notion of a Communication Control Block (CCB) cache. A CCB is a structure that contains the entire context associated with a connection. This includes the source and destination IP addresses and source and destination TCP ports that define the connection. It also contains information about the connection itself such

	as the current send and receive sequence numbers, and the first-hop MAC address, etc. The ***complete set of CCBs exists in host memory***, but ***a subset of these may be "owned" by the card*** at any given time. This subset is the ***CCB cache***. The INIC can own up to 256 CCBs at any given time."
storing said second flow identifier in said flow memory; and	See above.
comparing said stored first flow identifier and said stored second flow identifier.	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This ***hash*** is used to ***index*** directly into a ***hash table*** on the INIC that ***points to entries in a CCB header table***. The ***header table entries are chained on the hash table entry***. The microcode uses the hash to determine if a CCB exists on the INIC for this frame. It does this by following this chain from the hash table entry, and for each chained header table entry, ***comparing its source and destination addresses and ports with those of the frame***."
9. The method of claim 8, wherein said flow memory is an associative memory in said communication device.	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This ***hash*** is used to ***index*** directly into a hash table on the INIC that ***points to entries in a CCB header table***. The ***header table entries are chained on the hash table entry***. The microcode uses the hash to determine if a CCB exists on the INIC for this frame. It does this by ***following this chain from the hash table entry***, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame."
10. The method of claim 3, further comprising storing said first packet in a packet memory.	(85:27-29) "As frames are received, they are ***placed into 2K-byte DRAM buffers*** by the Receive hardware sequencer, along with 16 bytes of the above frame status.
11. The method of claim 10, wherein said determining comprises comparing said first flow identifier configured to identify said first communication flow with a second flow identifier configured to identify a second communication flow comprising a packet stored in said packet memory.	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This ***hash*** is used to ***index*** directly into a ***hash table*** on the INIC that points to entries in a CCB header table. The ***header table entries are chained on the hash table entry***. The microcode ***uses the hash to determine if a CCB exists on the INIC for this frame***. It does this by following this chain from the hash table entry, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame."
12. The method of claim 3, further comprising informing said host computer of said transfer of said first packet.	(2:29-30) "Each of these segments may result in an interrupt to the CPU."
13. The method of claim 12, wherein said informing comprises configuring an indicator in a host	(7:39-8:4) "Fortunately, in the case of NetBIOS transactions (such as SMB), we are explicitly told the length of the session message in the NetBIOS header itself. With this we can simply ***indicate*** a small amount

memory.	of data to the host immediately upon receiving the first segment. The client will then ***allocate enough memory*** for the entire NetBIOS transaction, which we can then use to DMA the remainder of the data into as it arrives. In the case of a large (56k for example) NetBIOS session message, all but the first couple hundred bytes will be DMA'd to their final destination in memory."
14. The method of claim 13, wherein said indicator is configured to indicate that said host computer should delay processing said first packet until said second packet is transferred to said host computer.	(7:18-27) "NT has provided a mechanism by which a transport driver can "***indicate***" a small amount of data to a client above it while telling it that it has more data to come. The client, having then received enough of the data to know what it is, is then responsible for allocating a block of memory and passing the memory address or addresses back down to the transport driver, which is in turn responsible for moving the data into the provided location. We will make use of this feature by providing a small amount of any received data to the host, with a ***notification*** that we have more data pending. When this small amount of data is passed up to the client, and it returns with the address in which to put the remainder of the data, our host transport driver will pass that address to the INIC which will ***DMA the remainder of the data*** into its final destination."
15. The method of claim 13, wherein said indicator indicates that said host computer should not delay processing said first packet.	(7:28-36) "Clearly there are circumstances in which this does not make sense. When a small amount of data (500 bytes for example), with a push flag set indicating that the data must be delivered to the client immediately, it does not make sense to deliver some of the data directly while waiting for the list of addresses to DMA the rest. Under these circumstances, it makes more sense to deliver the 500 bytes directly to the host, and allow the host to copy it into its final destination. While various ranges are feasible, it is currently preferred that anything less than a segment's (1500 bytes) worth of data will be delivered directly to the host, while anything more will be delivered as a small piece (which may be 128 bytes), while waiting until receiving the destination memory address before moving the rest."
16. A method of transferring a packet from a network interface to a host computer, comprising: receiving a first packet at a network interface; storing said first packet in a packet memory; receiving a first flow identifier configured to identify a communication flow comprising said first packet;	See above. (85:27-29) "As ***frames are received***, they are ***placed into*** 2K-byte DRAM ***buffers*** by the Receive hardware sequencer, along with 16 bytes of the above frame status." See above. (86:1-5) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This ***hash*** is used to index directly into a ***hash table*** on the INIC that points to entries in a ***CCB header table***. The

	header table entries are chained on the ***hash table*** entry. (6:10-19) "This introduces the notion of a Communication Control Block (CCB) cache. A ***CCB*** is a structure that contains the entire ***context associated with a connection***. This includes the ***source and destination IP addresses and source and destination TCP ports that define the connection.***
storing said first flow identifier in a flow memory;	(86:3-5) "This ***hash*** is used to index directly into a ***hash table*** on the INIC that points to entries in a ***CCB header table.*** The header table entries are chained on the ***hash table*** entry."
searching said flow memory for a second packet in said communication flow received at the network interface after said first packet;	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This ***hash*** is used to ***index*** directly into a hash table on the INIC that ***points to entries in a CCB header table***. The ***header table entries are chained on the hash table entry***. The microcode uses the hash to determine if a CCB exists on the INIC for this frame. It does this by following this chain from the hash table entry, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame."
transferring said first packet to said host computer; and	(5:25-27) "In the slow-path case, ***network frames are handed to the system*** at the MAC layer and passed up through the host protocol stack like any other network frame."
configuring an indicator in a host memory to indicate whether processing of said first packet by said host computer should be delayed to await transfer of said second packet to said host memory.	(7:18-27) "NT has provided a mechanism by which a transport driver can "***indicate***" a small amount of data to a client above it while telling it that it has more data to come. The client, having then received enough of the data to know what it is, is then responsible for allocating a block of memory and passing the memory address or addresses back down to the transport driver, which is in turn responsible for moving the data into the provided location. We will make use of this feature by providing a small amount of any received data to the host, with a ***notification*** that we have more data pending. When this small amount of data is passed up to the client, and it returns with the address in which to put the remainder of the data, our host transport driver will pass that address to the INIC which will ***DMA the remainder of the data*** into its final destination."
17. The method of claim 16, wherein said generating comprises: receiving an index of said communication flow in a flow database; wherein said flow identifier	See above. (86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports.*** This hash is used to ***index*** directly into a hash table on the INIC that points to entries in a ***CCB header table***. The

comprises said index.	header table entries are chained on the hash table entry. The microcode uses the hash to determine if a CCB exists on the INIC for this frame. It does this by following this chain from the hash table entry, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame."
18. The method of claim 16, wherein said receiving comprises: receiving a flow key comprising an identifier of a source of said first packet and an identifier of a destination of said first packet; wherein said flow identifier comprises said flow key.	See above. (6:17-19) "CCBs are initialized by the host during TCP connection setup. Once the connection has achieved a "steady-state" of operation, its associated ***CCB*** can then be ***turned over to the INIC***, putting us into fast-path mode." (6:10-12) "This introduces the notion of a Communication Control Block (CCB) cache. A CCB is a structure that contains the entire context associated with a connection. This includes the ***source and destination IP addresses and source and destination TCP ports*** that define the connection."
19. The method of claim 16, wherein said packet memory comprises said flow memory.	(86:18-22) "Get the ***CCB*** into an SRAM CCB buffer; there are 16 of these buffers in SRAM and they are not flushed to ***DRAM*** until the buffer space is needed by other CCBs. Acquisition and flushing of these CCB buffers is controlled by a hardware LRU mechanism. Thus getting the CCB into a buffer may involve flushing another CCB from its SRAM buffer."
20. The method of claim 16, wherein said configuring comprises: storing a first indicator in a host memory if said communication flow comprises said second packet; and storing a second indicator in said host memory if said first packet is the only packet in said packet memory that is part of said communication flow.	See above. (86:7-8) "If a ***match is found***, then the frame will be processed against the CCB by the INIC." (5:27-28) "In the fast path case, ***network data is given to the host*** after the headers have been processed and stripped." (5:25-27) "In the slow-path case, network frames are handed to the system at the MAC layer and passed up through the host protocol stack like any other network frame." (83:8-9) "When the INIC receives a frame, one of its immediate tasks is to determine if the frame is for a CCB that it controls. If not, ***the frame is passed to the host*** on what is termed the slow path." (13:17-18) "As mentioned above, the fast-path flow puts a ***header*** into a ***header buffer*** that is then forwarded to the host."
21. A computer system for processing a packet received from a network interface, comprising: a network interface configured to receive a first packet from a	(3:25-26) "Alacritech was formed with the idea that the network processing described above could be offloaded onto a cost-effective Intelligent Network Interface Card (INIC)." (6:30-31) "When ***a frame is received by the INIC***, it must verify it completely before it even determines whether it belongs to one of its

network and transfer said first packet to a host computer memory, said network interface comprising:	CCBs or not."
a packet memory configured to store said first packet;	(85:27-28) "As frames are received, they are ***placed into*** 2K-byte DRAM ***buffers*** by the Receive hardware sequencer."
a flow memory for storing a first flow number associated with said first packet, wherein said first flow number is configured to identify a communication flow comprising said first packet;	(24:35-39) "The initial command from ATCP to INIC expresses an "intention" to hand out the context. It carries a ***context number, context numbers*** are allocated by the ATCP driver, which ***keeps a per-INIC table of free and in-use context numbers***. It also includes the source and destination IP addresses and ports, which will allow the INIC to establish a "provisional" context." (6:10-16) "This introduces the notion of a Communication Control Block (CCB) cache. A CCB is a structure that contains the entire context associated with a connection. This includes the source and destination IP addresses and source and destination TCP ports that define the connection. It also contains information about the connection itself such as the current send and receive sequence numbers, and the first-hop MAC address, etc. The ***complete set of CCBs exists in host memory***, but a ***subset of these may be "owned" by the card*** at any given time. This ***subset is the CCB cache***. The INIC can own up to 256 CCBs at any given time."
a packet batcher configured to determine whether the communication flow includes a second packet stored in said packet memory after said first packet; and	(86:1-8) "The receive sequencer has already generated a hash based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This hash is used to index directly into a hash table on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The ***microcode uses the hash to determine if a CCB exists*** on the INIC for this frame. It does this by following this chain from the hash table entry, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame."
a notifier configured to:	(169:2-5) "The INIC includes special hardware assist for the implementation of message and pointer queues. The hardware assist is called the queue manager (Qmg) and manages the movement of queue entries between Cpu and Sram, between dma sequencers and Sram as well as between Sram and dram."
store a first code in a host indicator if said packet memory includes the second packet; and	(86:1-5) "If ***bit 29 is not set***, then there may be an onboard CCB for this frame. The receive sequencer has already generated a hash based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This hash is used to index directly into a hash table on the INIC that points to entries in a CCB header table. The header

	table entries are chained on the hash table entry." (86:5-9) "The microcode uses the hash to determine if a CCB exists on the INIC for this frame. It does this by following this chain from the hash table entry, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame. If a match is found, then the frame will be processed against the CCB by the INIC. If not, then the frame is sent for slow-path processing."
store a second code in said host indicator if said packet memory does not include the second packet; and	(85:38-41) "If ***bit 29 is set***, this frame is going slow-path. Effectively this means that the frame will ***not be processed against an on-INIC CCB***. It will be passed directly to the host, although if the frame is TCPIP, then its checksums have already been validated by the hardware. Also, all other header validations have been performed."
a processor for processing a header portion of said first packet.	(110:2-3) "The ***processor*** is a convenient means to provide a programmable state-machine which is ***capable of processing incoming frames***."
22. A computer readable storage medium storing instructions that, when executed by a computer, cause the computer to perform a method of transferring a packet from a network interface to a host computer, the method comprising:	(110:12) "The ***processor instructions reside in the on-chip control-store, which is implemented as a mixture of ROM and Sram***." (80:6-31) "As specified in other sections, the INIC supplies a set of 3 custom processors (CPUs) that provide considerable hardware-assist to the microcode running thereon... • Multiple register contexts or process slots with register access controlled by simply setting a process register . The Protocol Processor will provide 512 SRAM-based registers to be shared among the 3 CPUs in any way desired. The current implementation uses 16 processes of 16 registers each, leaving 256 scratch registers to be shared. A set of CPU-specific registers that are the same local-cpu register number, but for which the real register is determined by an offset based on the CPU number; this allows multiple CPUs to execute the same code at the same time without register clashes or interlocks. These registers are a part of the above-mentioned scratch pool."
receiving a first packet at a communication device;	(80:12-13) "***Receive Frame Processing*** Receive-frame processing can be broken down into the following stages"
identifying a first communication flow comprising said first packet with a first flow identifier configured to identify both the source entity and the destination entity;	(6:10-14) "CCBs are ***initialized*** by the host during ***TCP connection setup***." (6:10-12) "A ***CCB*** is a structure that contains the entire ***context*** associated with a ***connection***. This includes the ***source and destination IP addresses and source and destination TCP ports*** that ***define the connection***. (4:16-19) "The ... ***context*** is ... ***identified by the IP source and destination addresses and TCP source and destination ports***."

determining whether said first communication flow also comprises a second packet received at said communication device after said first packet was received at said communication device; and	(86:1-8) "The receive sequencer has already generated a hash based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***. This hash is used to index directly into a hash table on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The microcode uses the hash to ***determine if a CCB exists on the INIC for this frame***. It does this by following this chain from the hash table entry, and for each chained header table entry, ***comparing its source and destination addresses and ports with those of the frame***."
transferring said first packet to a host computer for processing in accordance with a communication protocol associated with said first packet.	(5:25-27) "In the slow-path case, ***network frames are handed to the system*** at the MAC layer and ***passed up through the host protocol stack*** like any other network frame."
23. A processor readable storage medium containing a data structure configured to store information concerning a packet to be transferred from a network interface to a host computer, the data structure including one or more entries, each entry comprising:	(80:6-7) "As specified in other sections, the INIC supplies a set of 3 custom processors (CPUs) that provide considerable hardware-assist to the microcode running thereon." (6:10-16) "A CCB is a structure that contains the entire context associated with a connection. This includes the source and destination IP addresses and source and destination TCP ports that define the connection. It also contains information about the connection itself such as the current send and receive sequence numbers…"
a flow number configured to identify a communication flow comprising a first packet received at the network interface from a source entity for a destination entity associated with the host computer; and	(24:35-39) "The initial command from ATCP to INIC expresses an "intention" to hand out the context. It carries a ***context number***; ***context numbers*** are allocated by the ATCP driver, which ***keeps a per-INIC table of free and in-use context numbers***. It also includes the source and destination IP addresses and ports, which will allow the INIC to establish a "provisional" context." (86:2-4) "a ***hash*** based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***. This hash is used to index directly into a ***hash table*** on the INIC that points to ***entries in a CCB header table***."
a validity indicator configured to provide:	(85;15-28) "Much ***header validation*** is implemented in hardware in conjunction with MAC processing by scanning the data as it flies by. The following tests are performed and appropriate ***status is generated*** by the Receive hardware sequencer: • MAC header: determine if Ethernet/802.3, if MC/BC, if it matches our MAC address A or B, determine the network protocol, flag if not a MAC status of "good packet." • Network header: determine if header checksum is valid, header length is valid (e.g. IP >= 5), network length > header length, what the transport protocol is, if there is any

	fragmentation or network options, destination network address (one of ours?) • Transport header: checksum is valid (incl pseudo-header if relevant), header length is valid (e.g. TCP >= 5), length is valid, what is the session layer protocol (e.g. SMB, HTTP or FTP data), are there any transport flags set (e.g. FIN/SYN/URG/RST bits), and any options present. As frames are received, they are placed into 2K-byte DRAM buffers by the Receive hardware sequencer, along with 16 bytes of the above ***frame status***."
a first indication if said first packet is ready for transfer to the host computer; and	(85:38-41) "If bit 29 is set, this frame is going slow-path. Effectively this means that the frame will not be processed against an on-INIC CCB. It will be passed directly to the host, although if the frame is TCPIP, then its checksums have already been validated by the hardware. Also, all other header validations have been performed."
a second indication if said first packet is a control packet;	(5:8-14) "We have also opted to ***not handle TCP connection and breakdown***. Here is a list of other TCP "exceptions" which we have elected to not handle on the INIC: Retransmission Timeout – Occurs when we do not get an acknowledgement for previously sent data within the expected time period. Out of order segments – Occurs when we receive a segment with a sequence number other than the next expected sequence number. ***FIN segment – Signals the close of the connection.***" (52:24-25) "when we detect an incoming ***SYN segment*** in tcp_input(), we call ATKPassiveConnect(), which is located in the file atktdi.c."
wherein said data structure is searched for a second entry containing said flow number when said first packet is transferred to the host computer to determine if said communication flow also comprises a second packet received at the network interface after said first packet.	(86:1-8) "The receive sequencer has already generated a hash based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***. This hash is used to index directly into a hash table on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The microcode uses the hash to ***determine if a CCB exists on the INIC for this frame***. It does this by following this chain from the hash table entry, and for each chained header table entry, ***comparing its source and destination addresses and ports with those of the frame.***"
24. The method of claim 3, wherein said identifying comprises:	See above.
parsing said first packet to retrieve an identifier of the source entity and an identifier of the destination entity; and	(6:36-38) "The ***header*** is fully ***parsed*** by hardware and its type is summarized in a single status word. The checksum is also verified automatically in hardware, and a hash key is created out of the ***IP addresses and TCP ports*** to expedite CCB lookup."

combining said source entity identifier and said destination entity identifier to form said first flow identifier.	See above.
25. A communication interface, comprising:	
a header parser configured to parse a header of a first packet received at the communication interface, wherein the first packet was issued from a source entity for a destination entity;	(6:36-38) "The ***header*** is fully ***parsed*** by hardware and its type is summarized in a single status word. The checksum is also verified automatically in hardware, and a hash key is created out of the IP addresses and TCP ports to expedite CCB lookup."
a flow database configured to facilitate management of a communication flow comprising the first packet, the flow database comprising:	(6:10-16) "This introduces the notion of a Communication Control Block (CCB) cache... The ***complete set of CCBs exists in host memory***, but a subset of these may be "owned" by the card at any given time. This subset is the ***CCB cache***. The INIC can own up to 256 CCBs at any given time."
a flow key configured to identify the communication flow using identifiers of the source entity and the destination entity;	(6:10-16) "A ***CCB*** is a structure that contains the entire ***context associated with a connection***. This includes the ***source and destination IP addresses and source and destination TCP ports*** that define the connection. It also contains information about the connection itself such as the current send and receive sequence numbers..."
an activity indicator configured to indicate a recency with which a packet in the communication flow has been received; and	(6:10-16) "A CCB ... also contains information about the connection itself such as the ***current send and receive sequence numbers***..."
a validity indicator for indicating whether the communication flow is valid;	(85;15-28) "Much ***header validation*** is implemented in hardware in conjunction with MAC processing by scanning the data as it flies by. The following tests are performed and appropriate status is generated by the ***Receive hardware sequencer***: • MAC header: determine if Ethernet/802.3, if MC/BC, if it matches our MAC address A or B, determine the network protocol, flag if not a MAC status of "good packet." • Network header: determine if header checksum is valid, header length is valid (e.g. IP >= 5), network length > header length, what the transport protocol is, if there is any fragmentation or network options, destination network address (one of ours?) • Transport header: checksum is valid (incl pseudo-header if relevant), header length is valid (e.g. TCP >= 5), length is valid, what is the session layer protocol (e.g. SMB, HTTP or

	FTP data), are there any transport flags set (e.g. FIN/SYN/URG/RST bits), and any options present. As frames are received, they are placed into 2K-byte DRAM buffers by the Receive hardware sequencer, along with 16 bytes of the above ***frame status.***"
a code generator configured to generate an operation code for the first packet, to facilitate forwarding of the first packet toward the destination entity; and	(85:38-41) "If bit 29 is set, this frame is going slow-path. Effectively this means that the frame will not be processed against an on-INIC CCB. It will be passed directly to the host, although if the frame is TCPIP, then its checksums have already been validated by the hardware. Also, all other header validations have been performed."
a packet batching module configured to determine whether a second packet received at the communication interface is part of the communication flow.	(86:1-8) "The receive sequencer has already generated a hash based on the network and transport addresses, e.g., IP source and destination addresses and TCP ports. This hash is used to index directly into a hash table on the INIC that points to entries in a CCB header table. The header table entries are chained on the hash table entry. The microcode uses the hash to ***determine if a CCB exists on the INIC for this frame.*** It does this by following this chain from the hash table entry, and for each chained header table entry, comparing its source and destination addresses and ports with those of the frame."
26. A method of processing a packet through a communication interface, the method comprising:	(3:25-26) "the network processing described above could be offloaded onto a cost-effective Intelligent ***Network Interface*** Card (INIC)."
receiving a first packet from a network, wherein the first packet is part of a communication flow between a source entity and a destination entity;	(6:30-31) "When a ***frame is received*** by the INIC, it must verify it completely before it even ***determines whether it belongs to one of its CCBs*** or not." (6:10-11) "A ***CCB*** is a structure that contains the entire ***context associated with a connection.***"
determining whether a header portion of the first packet conforms to one of a set of communication protocols;	(85;16-26) "The following tests are performed and appropriate status is generated by the Receive hardware sequencer: • MAC header: determine if Ethernet/802.3, if MC/BC, if it matches our MAC address A or B, ***determine the network protocol***, flag if not a MAC status of "good packet." • Network header: determine if header checksum is valid, header length is valid (e.g. IP >= 5), network length > header length, ***what the transport protocol is***, if there is any fragmentation or network options, destination network address (one of ours?) • Transport header: checksum is valid (incl pseudo-header if relevant), header length is valid (e.g. TCP >= 5), length is valid, ***what is the session layer protocol*** (e.g. SMB, HTTP or FTP data), are there any transport flags set (e.g. FIN/SYN/URG/RST bits), and any options present."

assembling a flow identifier to identify the communication flow, wherein said flow identifier comprises a source entity identifier and a destination entity identifier;	(86:1-8) "The receive sequencer has already generated a ***hash*** based on the network and transport addresses, e.g., ***IP source and destination addresses and TCP ports***.
updating a flow database configured to facilitate management of communication flows through the communication interface, wherein said updating comprises:	(87:18-37) "Once the INIC is handling CCBs, i.e. fast-path processing, there are numerous other events that need to be processed apart from received frames for that CCB. The following are the relevant events: lock a new context (from Xmit); unlock a new context (from Xmit); receive frame (complete or incomplete) from the CCB queue; receive window update from the CCB queue; receive a partial/split NetB header from the CCB queue; end of the CCB queue of frames; flush context request from host; flush context request from Xmit; context release/flush complete from Xmit. The following summarizes Receive Event processing: Get control of the associated CCB; this involves locking the CCB to stop other processing (e.g. Transmit) from altering it while this processing is taking place. Get the CCB into an SRAM CCB buffer; If the event is "Check CCB queue", check the internal queue in the CCB; if there are frames queued, dequeue the next one, get its header into an SRAM header buffer and examine it to generate a specific event; if no frames are queued, exit; Either way, process the event against the CCB's FSM."
configuring a flow activity indicator associated with the communication flow to reflect receipt of the first packet; and	(6:10-16) "A ***CCB*** is a structure that contains the entire context associated with a connection. This includes the source and destination IP addresses and source and destination TCP ports that define the connection. It also contains information about the connection itself such as the ***current send and receive sequence numbers***..."
configuring a flow validity indicator associated with the communication flow to indicate that the communication flow is valid;	(80:13-17) "A Receive hardware sequencer that completely ***validates*** an input header as the frame is being received by the MAC, validates TCP and IP checksums, generates a frame status and a context lookup hash, moves the frame into a DRAM buffer and queues the ***frame address*** and ***status*** for processing by the Receive CPU into one of the hardware queues mentioned above."
assigning an operation code to the first packet, said operation code indicating whether a portion of data	(85:38-41) "If bit 29 is set, this frame is going slow-path. Effectively this means that the frame will not be processed against an on-INIC CCB. It will be passed directly to the host, although if the frame is TCPIP, then its

in the first packet is reassembleable with another portion of data in another packet in the communication flow; and	checksums have already been validated by the hardware. Also, all other header validations have been performed."
determining whether a second packet received at the communication interface is part of the communication flow.	(86:7-8) "If a ***match is found***, then the frame will be ***processed against the CCB*** by the INIC."
27. The method of claim 3, further comprising:	See above.
storing a first indicator in the host computer if said first communication flow comprises said second packet; and	(86:7-8) "If a match is found, then the frame will be processed against the CCB by the INIC." (5:27-28) "In the fast path case, network data is given to the host after the headers have been processed and stripped."
storing a second indicator in the host computer if said first packet is the only packet stored in the communication device that is part of said communication flow.	(5:25-27) "In the slow-path case, network frames are handed to the system at the MAC layer and passed up through the host protocol stack like any other network frame." (83:8-9) "When the INIC receives a frame, one of its immediate tasks is to determine if the frame is for a CCB that it controls. If not, ***the frame is passed to the host*** on what is termed the slow path." (13:17-18) "As mentioned above, the fast-path flow puts a ***header*** into a ***header buffer*** that is then forwarded to the host."

Applicants: Boucher et al.
Atty. Docket ALA-008H

CONCLUSION

In view of the above remarks, Applicants request that an interference be declared between the present application and U.S. Patent No. 6,483,804. If the Examiner would like to discuss any aspect of this application, including how the claims are supported by Applicants' specification, the Examiner is requested to call the undersigned.

Respectfully submitted,

Mark Lauer
Reg. No. 36,578
6601 Koll Center Plaza,
Suite 245
Pleasanton, CA 94566
Tel: (925) 484-9295
Fax: (925) 484-9291

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as Express Mail Label No. ER 264470458 US in an envelope addressed to MS Patent Application, Commissioner for Patents, P.O. Box 1450 Alexandria, VA 22313-1450, on October 3, 2003.

Date: 10-3-03

Mark Lauer